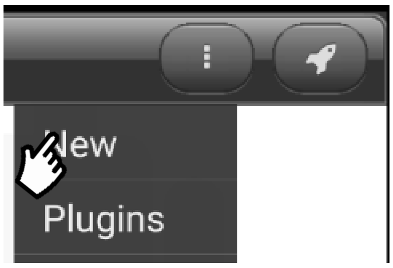
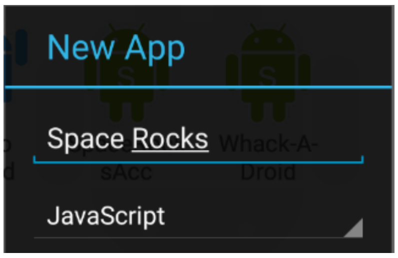
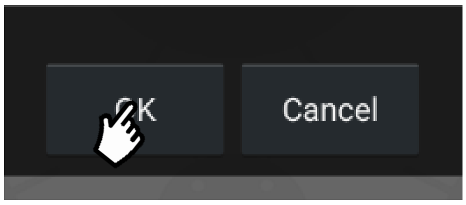
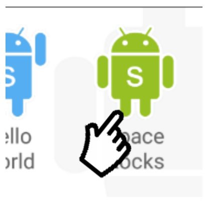
Space Rocks

Welcome to the Space Rocks Tutorial! Read on to learn about **Images, Backgrounds, Intervals, timers, plugins, sensors, the accelerometer, explosions** and **displaying a score**

Step 1 – Create your new App

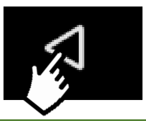
Create a new **JavaScript** App	Call it **Space Rocks**	Click **OK**
		



If you **Run** your new App by clicking the icon

You should see it says **Hello** on the screen.

Press back arrow to exit the app.



Step 2 – Set the Score

Open your App for editing by **long pressing the icon**, and selecting **Edit** from the Menu.
Find the word **"Hello"** and change it to **0**.
Add a new line of code below the line that says '`SetTextSize`'.
Your code should now look like this!

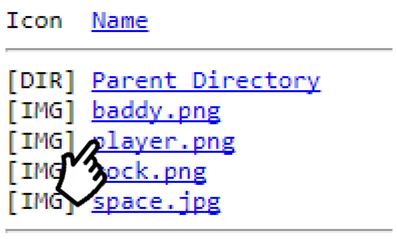
```
function OnStart()
{
	//Create a layout with objects vertically centered.
	lay = app.CreateLayout( "linear", "VCenter,FillXY" );

	//Create a text label and add it to layout.
	txt = app.CreateText( "0" );
	txt.SetTextSize( 32 );
	txt.SetTextColor( "yellow" );
	lay.AddChild( txt );

	//Add layout to app.
	app.AddLayout( lay );
}
```

Step 3 – Find some Images….

We need to download some **images** for our game

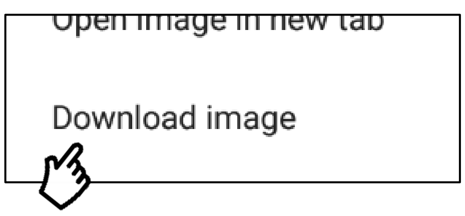
They can be found at - http://androidscript.org/tutorials/SpaceRocks/Images

Click on player.png

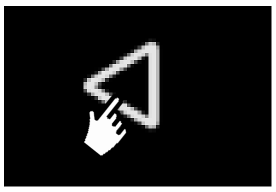


```
Icon   Name
[DIR]  Parent Directory
[IMG]  baddy.png
[IMG]  player.png
[IMG]  rock.png
[IMG]  space.jpg
```

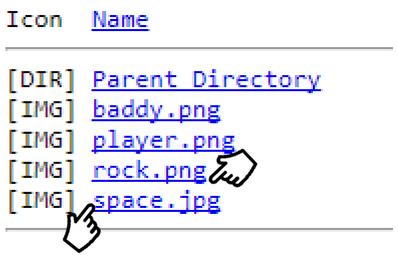
Press and Hold on the Picture and then click Download Image



Click Back.



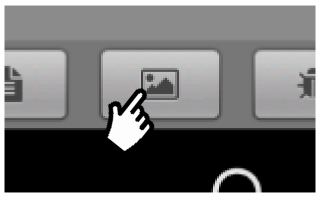
Now do the same for rock.png and space.jpg



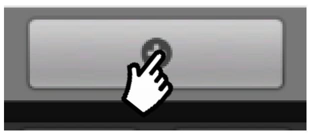
```
Icon   Name
[DIR]  Parent Directory
[IMG]  baddy.png
[IMG]  player.png
[IMG]  rock.png
[IMG]  space.jpg
```

Now we need to add them to our project.

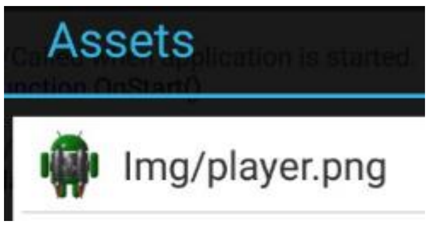
Click on the icon to go to the Assets section



Click the + button to Add Assets, then open your Downloads Folder



Add your player, then repeat these last 2 steps for rock and space



You should now see them in the Assets List.

Step 4 – Change the Background

Add a new **SetBackground** line below the line that says 'CreateLayout' like this:-

Now run your app.

```
function OnStart()
{
    //Create a layout with objects vertically centered.
    lay = app.CreateLayout( "linear", "VCenter,FillXY" );

    lay.SetBackground( "Img/space.jpg" );
```

Step 5 – Add a Player

Find the word **"Linear"** in your program and change it to **"Absolute".**

Type these 3 lines below the line that says 'AddChild'

Your complete code should now look like this:-

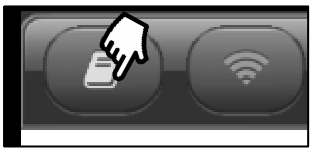
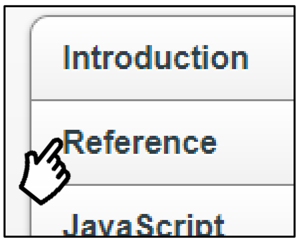
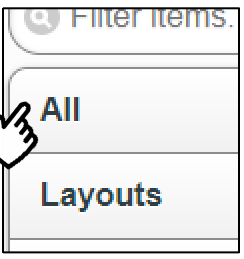
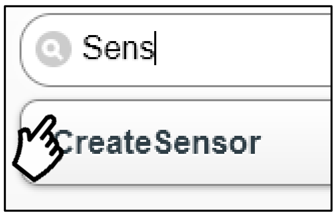
```
function OnStart()
{
    //Create a layout with objects vertically centered.
    lay = app.CreateLayout( "Absolute", "VCenter,FillXY" );

    lay.SetBackground( "Img/space.jpg" );

    //Create a text label and add it to layout.
    txt = app.CreateText( "Hello" );
    txt.SetTextSize( 32 );
    lay.AddChild( txt );

    //Create a player image.
    imgPlayer = app.CreateImage( "Img/player.png" );
    lay.AddChild( imgPlayer );


    //Add layout to app.
    app.AddLayout( lay );
}
```

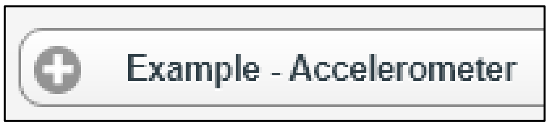
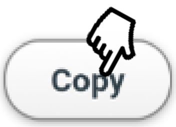
Now **Run your App**, You should see a player Image.

Step 6 – Adding the Sensor

We are going to use the **Accelerometer** to control our player!

To access this, we need to **create the sensor in our app**, then we can **read the values** to tell if the phone has been rotated!

Open the Docs	Select Reference	Click All	Type Sensor into the filter and Click the CreateSensor
			

Find the first example – Accelerometer	Grab the bold part by clicking Copy
	

Now paste the code into your **OnStart** function just below the line that says '**AddLayout**'

Like this :-

```
    //Add layout to app.
    app.AddLayout( lay );

    sns = app.CreateSensor( "Accelerometer" );
    sns.SetOnChange( sns_OnChange );
    sns.Start();
```

Now go back to the same **Accelerometer** example and click Copy All the **sns_OnChange** function from the example and **paste it into your code at the very bottom**. Then delete the **OnStart** function that came with it.

The bottom of your code should now look like this like this:

Once you have it in. **Run your App.** What happens when you rotate the phone, see if you can work out which axis we want to read

```
function sns_OnChange( x, y, z, time )
{
  txt.SetText( "x="+x + "\n y="+y + "\n z="+z );
}
```

Step 7 – Take Control and check your Code

Check the top of your code looks like this. We are adding a few bits to the bottom so watch out and make sure you don't miss them!

```
//Called when application is started.
function OnStart()
{
	//Create a layout with objects vertically centered.
	lay = app.CreateLayout( "Absolute", "VCenter,FillXY" );

	lay.SetBackground( "Img/space.jpg" );

	//Create a text label and add it to layout.
	txt = app.CreateText( "Hello" );
	txt.SetTextSize( 32 );
	lay.AddChild( txt );

	//Create a player image.
	imgPlayer = app.CreateImage( "Img/player.png" );
	lay.AddChild( imgPlayer );


	//Add layout to app.
	app.AddLayout( lay );

	sns = app.CreateSensor( "Accelerometer" );
	sns.SetOnChange( sns_OnChange );
	sns.Start();

	y = 0.7; x = 0.4; direction = 0;
}

function sns_OnChange( x, y, z, time )
{
    if (x > 0) direction = 1;
    else direction = 1;

    x += direction * 0.01;
    imgPlayer.SetPosition( x, y );
}
```

Firstly we need to set up a base line for our Player's **x and y** positions and the **direction**. So that when the game starts the player is **static** and in **the middle of the screen.**

Change the bottom of the **OnStart** function to look like this:-

Now let's make our **player move when the phone is tilted**

Change the sns_**OnChange** function to look like this:-

We get 3 pieces of info back from the sensor…

The x, y, and z co-ordinates of the phone.

We are only concerned with the x, we want to alter the direction depending on whether it is positive or negative.

What will happen to the variable direction when the phone is tilted?

Then we need to move the Player Image, we want it to move a tenth of the width of the screen, in the direction we set with the sensor

Add a this new function at the very bottom of your program:-

Run your App and see what happens when you tilt the phone.

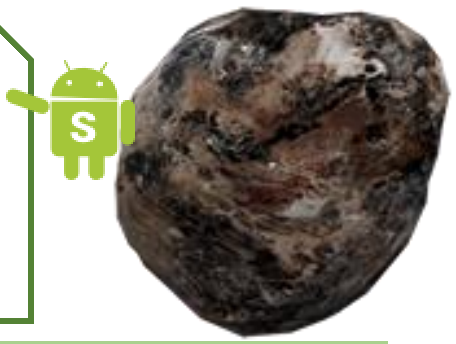
Step 8 – Add Some Rocks

Now we need to **create another timer** to **generate rocks** for our player to dodge, so add the following code to your `OnStart` function just **below the line** that says ‘`direction = 0`’:-

```
      rocks = [];
      rockTimer = setInterval( AddRock, 1000 );
```

Tip: When typing the name of a known variable try writing the first couple of letters and then pressing the <Alt> and <Space> keys at the same time. It will give you a list of variable names to choose from.

Create **a new function at the bottom of your code** called `AddRock` that will get called by our **rockTimer every 1000 milliseconds** like this:-



```
function AddRock()
{
    var size = 0.1 + 0.1 * Math.random();
    var imgRock = app.CreateImage( "Img/rock.png", size );
    rocks.push( imgRock );

    imgRock.x = Math.random();
    imgRock.y = 0;
    imgRock.SetPosition( imgRock.x, imgRock.y );

    lay.AddChild( imgRock );
}
```

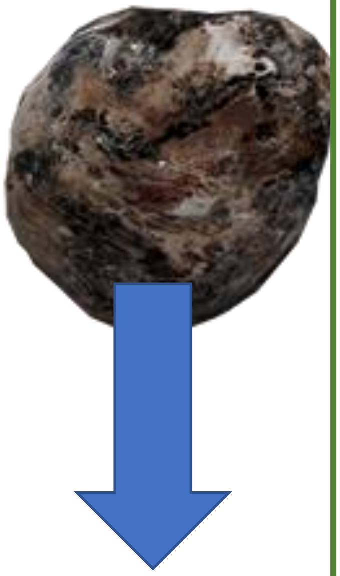
Now **run your app** and see what happens.

Step 9 – Rock Fall

We are going to use another timer to make our rocks fall down the screen.

Add this like to your `OnStart` function just **below the line** that says ‘`setInterval`:-

```
      rockFallTimer = setInterval( MoveRocks, 20 );
```



Next we need to make the **rocks start falling downwards**, so edit the ‘`MoveRocks`’ function to look like this:-

Then **Run your App**. Do the rocks fall?

What would happen if we added 0.02 to `rock.y` **instead?**

```
function MoveRocks()
{
    for( r in rocks )
    {
        rock = rocks[r];
        rock.y += 0.01;
        rock.SetPosition( rock.x, rock.y );

        if( rock.y > 1 )
        {
            lay.DestroyChild( rock );
            Arr.remove( rocks, rock );
        }
    }

}
```

Step 10 – Collision

At the moment the rocks just pass straight through our player, so we need to make this more interesting by detecting when **a rock collides (overlaps) with our player image**.

Type the following code into the ‘`MoveRocks`’ function just **below the line** that says ‘`rock.SetPosition`’:-

```
        rock.SetPosition( rock.x, rock.y );

        if( rock.IsOverlap( imgPlayer, 0.03 )  )
        {
            imgPlayer.Explode();

            app.ShowPopup( "Game Over" );
            clearInterval( rockFallTimer );
            clearInterval( rockTimer );
        }

        if( rock.y > 1 )
```

Now **run your app** and see what happens when your player touches a rock.

Step 11 – Update the Score

Add this line of code to the **OnStart** function **after the line** that says **setInterval**:-

```
score = 0;
```

Add these lines to the '**MoveRocks**' function just **below the line** that says '**Arr.remove**':-

Now **run your app**. You should see the score change when rocks pass by.

```
score++;
txt.SetText( score );
```

The bottom of your code should now look like this:-

```
function MoveRocks()
{
    for( r in rocks )
    {
        rock = rocks[r];
        rock.y += 0.01;
        rock.SetPosition( rock.x, rock.y );

        if( rock.IsOverlap( imgPlayer, 0.03 )  )
        {
            imgPlayer.Explode();

            app.ShowPopup( "Game Over" );
            clearInterval( rockFallTimer );
            clearInterval( rockTimer );
        }

        if( rock.y > 1 )
        {
            lay.DestroyChild( rock );
            Arr.remove( rocks, rock );
            score++;
            txt.SetText( score );
        }
    }

}

function AddRock()
{
    var size = 0.1 + 0.1 * Math.random();
    var imgRock = app.CreateImage( "Img/rock.png", size );
    rocks.push( imgRock );

    imgRock.x = Math.random();
    imgRock.y = 0;
    imgRock.SetPosition( imgRock.x, imgRock.y );

    lay.AddChild( imgRock );
}
```

Step 12 – Play the Game and Improve it.

You've made it the end of our tutorial.

We have looked at Images, Backgrounds, Intervals, timers, plugins, sensors, the accelerometer, explosions and displaying a score

Now play the game for a while and enjoy your hard work! While your playing though there are a few more things to think about. If you want to add some more stuff see the next page for our **Bonus Task**

- Is the game too easy or too hard? See if you can work out how to make it harder or easier by changing some of the numbers.
- Can you work out how to change the rock size?
- How about changing the speed of the rocks?
- Can you make your player larger?

Congratulations you are finished.

Well done!

Bonus Task

Task – Spin the Rocks

We need to create a property to store the spin angle of each rock, so add the this line of code to the 'AddRock' function, below the line that says 'imgRock.y ='

```
imgRock.y = 0;

imgRock.angle = 0;
```

Next we need to make the rock rotate a little each time the rocks move, so add the following two lines of code to the '**MoveRocks**' function, just below the line that says 'rock.SetPosition'.

```
rock.SetPosition( rock.x, rock.y );

rock.angle += 10;
rock.Rotate( rock.angle );
```

Now run your app. Do you see the rocks spinning?

Try adjusting the amount added to the angle and see what happens.

Can you work out how to make the spin rate of each rock random?